Interfaces Graphiques

Custom component

Jean-Baptiste.Yunes@univ-paris-diderot.fr
Université Paris Diderot

- *A Swing Architecture Overview*

The Inside Story on JFC Component Design

par Amy Fowler

- Pour personnaliser un composant il est nécessaire de prendre se préoccuper :
  - du rendu
    - un délégué de type `ComponentUI`
  - de la logique *métier*
    - un modèle
  - des services de base
    - `JComponent` + export partiel de la logique métier

- la plomberie du rendu
  - rappel : le rendu Swing est effectué non pas par le composant lui-même mais par délégation
- la classe `JComponent` fournit les services
  - `ComponentUI getUI()` / `setUI(ComponentUI)`
  - `updateUI()`
    - permet de capturer le moment où le `LookAndFeel` doit être pris en compte

- un `JComponent` ne se dessine pas lui-même
  - sinon comment pourrait-il prendre en compte une nouvelle apparence ?
    - emploi du motif conceptuel de la Délégation
      - à chaque `JComponent` est associé un objet s'occupant du rendu du composant
        - `ComponentUI`

- à un `JButton` est associé un `ButtonUI`
- à un `JLabel` est associé un `LabelUI`
- etc.

- modifiable par appel à
  - `setUI(ComponentUI)`
- consultable par appel à
  - `getUI(ComponentUI)`

Gestion du ComponentUI

- Un exemple : `PLAFUI.java`

- La logique métier est normalement (MVC) encapsulée dans une instance de Modèle
  - définir son propre modèle
  - il en existe de prédéfinis
    - `ButtonModel`
    - `ListModel`
    - etc.

- Exporter quelques services du modèle afin de rendre la vie plus facile
- et après tout, est-ce que le programmeur doit se préoccuper de ces histoires de modèle ?

Un exemple de composant personnalisé

- Un exemple : `Speedometer.java`